UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/713,943	11/14/2003	Bong Wan Kim	2013P125	4798

8791 7590 02/06/2007

BLAKELY SOKOLOFF TAYLOR & ZAFMAN
12400 WILSHIRE BOULEVARD
SEVENTH FLOOR
LOS ANGELES, CA 90025-1030

EXAMINER	
PATEL, KAUSHIKKUMAR M	
ART UNIT	PAPER NUMBER
2188	

SHORTENED STATUTORY PERIOD OF RESPONSE	MAIL DATE	DELIVERY MODE
3 MONTHS	02/06/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

If NO period for reply is specified above, the maximum statutory period will apply and will expire 6 MONTHS from the mailing date of this communication.

Office Action Summary	**Application No.** 10/713,943	**Applicant(s)** KIM ET AL.	
	Examiner Kaushikkumar Patel	**Art Unit** 2188	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *20 November 2006*.

2a)☒ This action is **FINAL**. 2b)□ This action is non-final.

3)□ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-20* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)□ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-20* is/are rejected.

7)□ Claim(s) _____ is/are objected to.

8)□ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)□ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *20 November 2006* is/are: a)☒ accepted or b)□ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)□ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☒ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☒ All b)□ Some * c)□ None of:

1.☒ Certified copies of the priority documents have been received.

2.□ Certified copies of the priority documents have been received in Application No. _____.

3.□ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) □ Notice of References Cited (PTO-892)

2) □ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) □ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.

4) □ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .

5) □ Notice of Informal Patent Application (PTO-152)

6) □ Other: _____.

DETAILED ACTION

Response to Amendment

1. This office action is in response to applicant's communication filed November 20, 2006 in response to PTO office action mailed July 20, 2006. The applicant's remarks and the amendments to the claims were considered with the results that follow.

2. In response to the last office action, claims 1, 3-9 and 12-19 have been amended. No new claims have been added. No claims have been deleted. As a result, claims 1-20 remain pending in this application.

3. The objections to drawings as well as the rejection of claims under 35 U.S.C. 112, second paragraph have been withdrawn due to amendments filed on November 20, 2006.

Response to Arguments

4. Applicant's arguments with respect to claims 1-20 have been considered but are moot in view of the new ground(s) of rejection.

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 1-3, 5-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over McMahon et al. (US 5,784,699) (McMahon herein after) and further in view of Donahue et al. (Hardware Support for Fast and Bounded-Time Storage Allocation, March 22, 2002) (Donahue herein after), [and Trainin et al. (US 2002/0144073 A1), Data Structures and Algorithms by John Morris (doc. 1) and Linked List Basics by Nick Parlante, introduced for evidentiary references (doc. 2)].

7. As per claim 1, McMahon teaches:

a data memory, which comprises a plurality of data blocks, each of which comprises a plurality of sub data, blocks having predetermined size (McMahon col. 2 line 65 – col. 3 line 1, where the slots are equivalent to plurality of data blocks in claim 1), and when there is a request for allocating memory space of variable size, allocates memory space in units of any one of the sub data blocks and the data blocks (col. 3 lines 9-12);

a free list which manages a free memory space of the data memory as an entry of a plurality of entries (col. 3 lines 1-4).

McMahon fails to teach use of pointers and registers. Donahue teaches tracking free lists using doubly-linked list using pointers and use of registers to store pointers (Donahue page 6, sec. 31. and sec. 3.2). Donahue explicitly failed to teach head and tail pointers but it is well known in the art that doubly-linked list inherently contains head and tail pointers (see doc.1, page 2 and doc. 2, page 26, the advantage is scanning can be performed in any direction and nodes can be added or deleted to any location in the list). Donahue further teaches multiple free lists each having their respective size

(Donahue, fig. 10, and page 6, left col. Sec. 3.1, "the heap is segregated by size into lists for relevant powers of two" and "free blocks are maintained in a doubly-linked list"), thus Donahue teaches multiple head and tail location (each size represented with doubly-linked list) and since memory-allocation is performed by traversing particular list or by dividing higher size blocks (Donahue, page 4, right col. "Buddy System Allocation"), thus Donahue teaches first and second head location to allocate different size of data block.

It would have been obvious to one having ordinary skill in the art at the time of the invention to use registers and pointers as taught by Donahue in the memory allocation system of McMahon to implement allocation in hardware for real time systems to improve system performance (see Donahue abstract).

As per claim 2, Donahue teaches registers to store addresses, which inherently teaches address translation (see Donahue page 6, sec. 3.2 and fig. 11 on page 7).

As per claim 3, McMahon teaches, the number of entries of the free list memory is the same as the number of entries of the data memory and the entries of the free list memory and the entries of the data memory have 1:1 corresponding relationship (McMahon col. 6 line 65 – col. 7 line 2, each free list has a mapping to each block that indicates the availability status of the block. Each block size has a free list making the mapping a bijection).

As per claim 5, McMahon teaches, the data memory has a hierarchical structure, in which the data memory contains a plurality of data sub blocks each having memory space of n bytes when n is a power of 2 and i and j are positive integers (i<j) (McMahon

col. 2 line 66 – col. 3 line 1), and each data block comprises a plurality of sub data blocks each having a memory space of $\frac{n}{2^i}$ –bytes and each sub data block comprises a plurality of sub data blocks each having a memory space of $\frac{n}{2^j}$ –byte (McMahon col. 6 – Table 1 row entries 1 and 2 teach block sizes of 16 bytes and 32 bytes, respectively, thus satisfying the structure of claim 5 limitations). (Donahue also teaches free lists with different size classes of relevant powers of two, see page 5, fig. 10 with different blocks and sub blocks having power of two, and page 6, sec. 3.1, satisfying the structure of claim 5 limitations).

As per claims 6 and 7, McMahon and Donahue teaches, each entry forming the free list memory comprises:

a plurality of bit masks each indicating whether or not sub data block is in use (McMahon col. 3 lines 4-6 and col. 6 line 65 – col. 7 line 2 and col. 7 lines 24-28 collectively describe a bijective mapping between bit masks and sub data blocks indicating availability status, Donahue also teaches free lists with respective blocks and sub blocks maintained with doubly-linked list pointers and bits masks to indicate block is currently allocated or not, see Donahue sec. 3.1, page 6). As pointers indicating entries of next and previous blocks/sub blocks and updating of pointers relative to allocating and deallocating memory is an inherent feature of lists maintained with doubly-linked lists (claim 7) (Doc. 1 and Doc. 2 teaches how pointers are used in memory allocation).

As per claim 8, McMahon teaches different size class of blocks with corresponding free lists and each list is capable of allocating memory according to status of bit mask value and their respective sizes (McMahon col. 3 line 4-6, col. 5 lines 32 – col. 6 line 40 and col. 6 line 65 – col. 7 line 2 and col. 7 lines 24-28, table on col. 6 shows different size memory lists and each able to allocate their respective size memory requirement satisfying structure of claim 8, Donahue also teaches multiple free lists with sizes of power of two capable of allocating their respective size requests, see fig. 10 on page 5, and page 4, sec. 3).

As per claim 9, the understanding of registers containing pointer pairs with respect to size classes of blocks and sub data blocks on specification page 6, line 18 – page 7 line 10 and fig.1 and fig. 2, states that each size class is tracked with their respective head and tail pointers (e.g. with respect fig. 1, 1HPFL and 1TPFL for size class of n –bytes, 2HPFL and 2TPFL for size class of $\frac{n}{2}$ –bytes and 3HPFL and 3TPFL for size class of $\frac{n}{4}$ –bytes. By dividing memory block of size n with power of two creates two size classes, if four equal sized blocks are created than it creates 3 size classes and eight equal size creates 4 allocatable size classes and so on. The equation (1 + $\log_2 X$) provides number of available size class with data block with equally sized sub data blocks, for example 256 –byte data block sub divided in four 64 –byte size creates 3 size class of 256 –bytes, 128 –bytes and 64 –bytes, which requires 3 pointer pairs, similarly if block is divided in eight equal 32 –bytes sub block creates four size classes of 256 –bytes, 128 –bytes, 64 –bytes and 32 –bytes and hence requires 4 pointer pairs. Donahue teaches binary buddy allocation method and creating different size class of sub blocks with their respective free list maintained with doubly-linked list (see fig. 10, page 5 and sec. 3.1 on page 6), thus Donahue inherently teaches limitations of claim 9.

As per claims 10 and 11, Donahue teaches allocating memory of size 2^k by searching the free list at index k and if size 2^k is available than allocates or divides blocks with size 2^{k+1} into two 2^k size blocks allocating one and adding other to free list of size 2^k (see Donahue page 4, col. 2. sec. 3, Buddy System Allocation). Thus Donahue satisfies limitations of claims 10 and 11.

As per claims 12 and 13, Donahue teaches deallocation of sub blocks and combining the deallocated block with its buddy to coalescing into larger block (see Donahue page 4, col. 2, sec. Buddy system deallocation. For further explanation of Buddy system, Trainin teaches memory allocation and deallocation in Buddy systems, See paragraphs [0010]-[0014], the advantage is that a block being freed can be found by a simple address calculation, see Donahue page 4, col. 2, sec. Buddy system Deallocation, page 6, sec. 3.1 and Trainin paragraph [0013]).

As per claims 14-16, Donahue teaches allocating memory request of size $\frac{n}{2^i}$ – bytes (or 2^k as per Donahue) by searching its respective free list index at k, and allocates the block if it is free, if the respective size is not free than larger size of 2^{k+1} divided into two 2^k size blocks allocating one and adding other to free list of size 2^k (see Donahue page 4, col. 2. sec. 3, Buddy System Allocation). It is inherent that block size smaller than requested size can not be allocated so as per limitation of step (a) of claim 14, inherently requires allocating available next larger size and for limitation of step (b), Donahue teaches dividing larger block into two equal size and allocating one and adding other to its respective free list (as explained on page 4 of Donahue). As per limitations of claims 15 and 16, since Donahue teaches managing free lists with doubly-linked list and free/busy bit, inherently requires setting of bit values and changing of head and tail pointers to their respective locations (see doc.1, page 2 and doc. 2, page 26, the advantage is scanning can be performed in any direction and nodes can be added or deleted to any location in the list. Doc.1 and doc. 2 also teaches managing respective pointers). Donahue further teaches multiple free lists each having their respective size (Donahue, fig. 10, page 6, left col. Sec. 3.1, "the heap is segregated by size into lists for relevant powers of two" and "free blocks are maintained in a doubly-linked list"), thus Donahue teaches multiple head and tail location (each size represented with doubly-linked list) and since memory-allocation is performed by traversing particular list or by dividing higher size blocks (Donahue, page 4, right col. "Buddy System Allocation"), thus Donahue teaches first and second head location to allocate different size of data block.

As per claims 17-19, Donahue teaches deallocating respective memory sizes and combining with their respective buddies (neighboring block as in present application) to coalescing into larger block (see Donahue page 4, Buddy system deallocation also on page 6, sec 3.1 paragraph 7). Similarly updating of free/used flag and pointers are inherent features as explained above. Donahue further teaches multiple free lists each having their respective size (Donahue, fig. 10, page 6, left col. Sec. 3.1, "the heap is segregated by size into lists for relevant powers of two" and "free blocks are maintained in a doubly-linked list"), thus Donahue teaches multiple head and tail location (each size represented with doubly-linked list) and since memory-allocation is performed by traversing particular list or by dividing higher size blocks (Donahue, page 4, right col. "Buddy System Allocation"), thus Donahue teaches first and second head location to allocate different size of data block.

As per claim 20, McMahon teaches implementing allocation algorithm as a program product (See McMahon col. 16 lines 12-30).

Claim Rejections - 35 USC § 103

8. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

9. Claim 4 is rejected under 35 U.S.C. 103(a) as being unpatentable over McMahon et al. (US 5,784,699) (McMahon herein after) and Donahue et al. (Hardware Support for Fast and Bounded-Time Storage Allocation, March 22, 2002) (Donahue herein after) as applied to claims 1-3 above and further in view of Murdocca et al. (Principles of Computer Architecture).

As per claim 4, McMahon and Donahue explicitly fail to teach calculating and address value. But as explained by Murdocca that memory is divided into pages with page frame numbers (see page 270, fig. 7-21). It can be seen that starting address of each page frame (entry) is calculated by "page frame (entry) number x page size (block size) + starting address of the memory". For example starting address of page frame 2 can be calculated by 2 x 1024 + 0 = 2048. It would have been obvious to one having ordinary skill in the art at the time of the invention would have calculated the starting address of each entry as evident from Murdocca in the system of McMahon and Donahue.

Conclusion

Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Kaushikkumar Patel whose telephone number is 571-272-5536. The examiner can normally be reached on 8.00 am - 4.30 pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Hyung Sough can be reached on 571-272-6799. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Kaushikkumar Patel
Examiner
Art Unit 2188

kmp

HYUNG SOUGH
SUPERVISORY PATENT EXAMINER

2-01-07